1 Оперативные данные тезисно:

Вам как опытному криптографу пришел секретный заказ от Архитектора. Он рассказал вам, что у него появилась новая «низко-энтропийная» система выписывания чеков, но так как он только на стадии разработки матрицы 1.0, то система Архитектора имеет некие уязвимости, которыми вам и предстоит найти и сообщить о них Архитектору. Ваша цель найти уязвимости в системе и подписать чек на ≥ 1 со счета Банка(Архитектора).

2 Описание работы системы:

Банк занимается выписыванием чеков. Любой клиент банка может выписать чек со своего счета на любое имя. Это означает, что клиент передает в распоряжение другого человека деньги со своего счета. Для того, что бы чек нельзя было подделать, компьютеры Архитектора подписывают каждый чек электронной подписью основанной на схеме Шнорра.

Кроме того, для того что бы люди которые принимают чек в качестве оплаты могли убедиться в том что чек оригинальный, а не *подделка*, банк предоставляет пользователям возможность проверки подписи чека на корректность.

3 Постановка задачи:

Вы являйтесь клиентом банка Архитектора. У вас есть две опции:

- Послать выписанный чек на подпись. Чек может быть выписан на любое имя (под именем понимается произвольная строка - имя человека кому вы выписывайте чек, не обязательно реальное имя.) С любой суммой, которая будет списываться с вашего счёта.
  В приложении банка это раздел ***Выписать чек***.
  В качестве чека вам вернется сообщение вида:

  $$\{ \; 's1' : s1, \; 's2' : str(s2), \; 'M' : M \; \}$$

  Закодированное алгоритмом: b64encode.
  M - чек составленный клиентом банка, который имеет вид:
  $(\underbrace{\text{счёт отправителя}}_{\text{32 бита}} || \underbrace{\text{сумма перевода}}_{\text{32 бита}})$, где $||$ - нужно понимать как конкотенацию строк.
  По умолчанию, в счет отправителя вставляется ваш личный счет клиента банка который имеет вид: $(SB : XXXXXXXX)$, где X - 16-я цифра. $S1, S2$ вычисляются как написано в 5

- Вторая опция - проверить чек на корректность подписи. Вы отправляйте в систему закодированное b64encode сообщение которое должно иметь вид как в предыдущем пункте, и далее проходит проверка подписи на корректность алгоритмом описанным в 5.
  В приложении банка это раздел ***проверить чек***.

От вас требуется послать закодированное b64encode сообщение, содержащее выписанный чек со счета Банка на более чем 1.

4 Генерация параметров подписи

Для того, чтобы банк мог подписывать сообщения ему необходимо сформировать параметры подписи в том, числе приватный и публичный ключи.

- На стороне банка выбирается простое число p
- Затем выбирается простое число q такое что $(p-1) \mod q = 0 (q$ делит $p-1)$
- Подбирается число $g \neq 1, g^q = 1 \ (mod \ p)$
- Генерируется секретный ключ $w < q$
- Генерируется публичный ключ проверки подписи $y = g^{q-w} \ (mod \ p)$
- В итоге приватный ключ это - $private\_key = w$, открытый ключ $public\_key = (p, q, g, y)$.

Публичный ключ является частью открытого ключа.

5 Система подписи

Пусть M - чек составленный клиентом банка. Вид чека: 3

Тезисно Подписание документа M осуществляется следующим образом:

- Формируется подпись чека M:
  $(S_1, S_2) = Snorr\_sign(message, private\_key)$
  Значения S_1, S_2 - это подпись.
- После любой желающий может проверить корректность подписи с помощью публичного ключа: $bool \ is\_correct = check\_Shnorr\_sign(M, S_1, S_2, public\_key)$
  $is\_correct = True$ если подпись корректно и $False$ иначе. Если подпись корректна, то чек валидный. Иначе чек считается не действительным.

Подробно

Подписание чека происходит следующим образом (*Shnorr_sig*):
Пусть M - текст самого чека

- Для каждого чека генерируется рандомное число с помощью функции $r = generator()$;
- Вычисляется $x = g^r \ (mod\, p)$
- Текст чека конкотенируется с десятичным представление числа x и все это хешируется с помощью хэш-функции SHA_{256}: $S1 = SHA_{256}(M||x)$
- Далее банк вычисляет S_2:
  $S_2 = r + w \cdot S1 \ (mod\, q)$. *Обратите внимание что S_2 вычисляется по модуля q, а не по модулю p.*
- Итоговое подписанное сообщение имеет вид: (M, S_1, S_2).

Проверка подписи в банке (*check_Shnorr_sign*):
Пусть (M, S_1, S_2) - подписанное сообщение, подпись которого необходимо проверить. Для этого банк совершает следующие шаги:

- формирует значение: $X = g^{S_2} \cdot y^{S_1} \ (mod\, p)$
- проверяет корректность равенства $H(M|X) = S_1$, и если это верно, то подпись принимается и чек вступает в силу, иначе чек отклоняется.

Известная информация:

- SHA_{256} в качестве криптографической функции H.
- $public\_key = (p, q, g, y)$, где:
  $p = 26176336699850542011792293720207749868 9$
  $q = 124901341482666746105456 3$
  $g = 234571634339194295184736054687977117090$
  $y = 241565911437637228086482251229669244 57.$

6 Цель:

Вам необходимо сформировать:

- чек $M = (\underbrace{\text{SB:FFFFFFFF}}_{\text{32 бита}} || \underbrace{\text{сумма перевода}}_{\text{32 бита}})$, где сумма перевода должна быть больше, либо равна 1;
- корректную подпись S_1, S_2 к чеку M;
- отправить эти данные в банк и получить подтверждение подписи.